Eur. Phys. J. C (2017) 77:188
DOI 10.1140/epjc/s10052-017-4755-3

THE EUROPEAN PHYSICAL JOURNAL C

Regular Article - Experimental Physics

Online production validation in a HEP environment

T. Harenberg[1], **T. Kuhl**[2], **N. Lang**[1], **P. Mättig**[1], **M. Sandhoff**[1,a], **C. Schwanenberger**[3], **F. Volkmer**[1]

[1] Bergische Universität, Gaußstr. 20, 42119 Wuppertal, Germany
[2] DESY Zeuthen, Platanenallee 6, 15738 Zeuthen, Germany
[3] DESY Hamburg, Notkestr. 85, 22607 Hamburg, Germany

Received: 1 December 2016 / Accepted: 14 March 2017 / Published online: 23 March 2017

Abstract In high energy physics (HEP) event simulations, petabytes of data are processed and stored requiring millions of CPU-years. This enormous demand for computing resources is handled by centers distributed worldwide, which form part of the LHC computing grid. The consumption of such an important amount of resources demands for an efficient production of simulation and for the early detection of potential errors. In this article we present a new monitoring framework for grid environments, which polls a measure of data quality during job execution. This online monitoring facilitates the early detection of configuration errors (specially in simulation parameters), and may thus contribute to significant savings in computing resources.

1 Introduction

Today's particle physics experiments produce vast amounts of data from the direct observation of particle interactions and from the computer simulations of those interactions. Simulations [1] are used to compare theoretical expectations to measured data, and require a detailed description of the fundamental physics processes involved and of the interaction between the detector's material and the particles produced in those processes. The production of simulations demands significant amounts of computing resources [2], which are provided by a worldwide network of computing centers in what is known as the worldwide large hadron collider (LHC) computing grid (wLCG). In the year 2012, the ATLAS experiment at the LHC simulated nearly ten trillion events,[1] stored in more than 54 PB of disk space and processed more than a 100,000 jobs every day. Figure 1 shows the monthly distribution of computing jobs.[2] Any job failure in processing such a vast amount of simulation would be a waste of CPU resources and should be avoided.

The complex configuration of simulation programs is at odds with the pressing need to promptly produce large samples of simulated events. Therefore, in the compromise between the need to produce samples and the complexity of the task at hand, there is a potential for introducing detrimental configuration errors. Simulation programs are configured via a large number of settings representing physics processes, particle decay modes, theoretical models and detector states at various energy levels and conditions.

Misconfigurations of the simulation may lead to job aborts in the best of cases, which are easy to detect and recover. However, some faulty settings do not cause jobs to terminate with errors, but instead, they lead to results which are useless or misleading for physics analyses. The complexity of simulations and the typically large dimensions of parameter spaces frequently conspire to mask faulty configurations, which are often not discovered until large samples have already been produced and used extensively in physics analyses. This constitutes an unfortunate waste of limited human and computing resources, and an obstacle to the efficient production of prompt physics results.

In order to reduce the number of failures during the production of simulated events, the quality of the data should be monitored during job execution (online monitoring), and a system of automated alerts triggered by common problems should be implemented.

In this paper, we describe a tool that meets these requirements. The job execution monitor (JEM) [3,4] is a job-based monitoring framework currently used by the ATLAS Collaboration to monitor the production of simulation samples, and

[1] Each event represents the detector's information on all particles produced in a single particle collision.

[2] A job consists of a collection of programs processing the same physics process.

[a] e-mail: sandhoff@uni-wuppertal.de

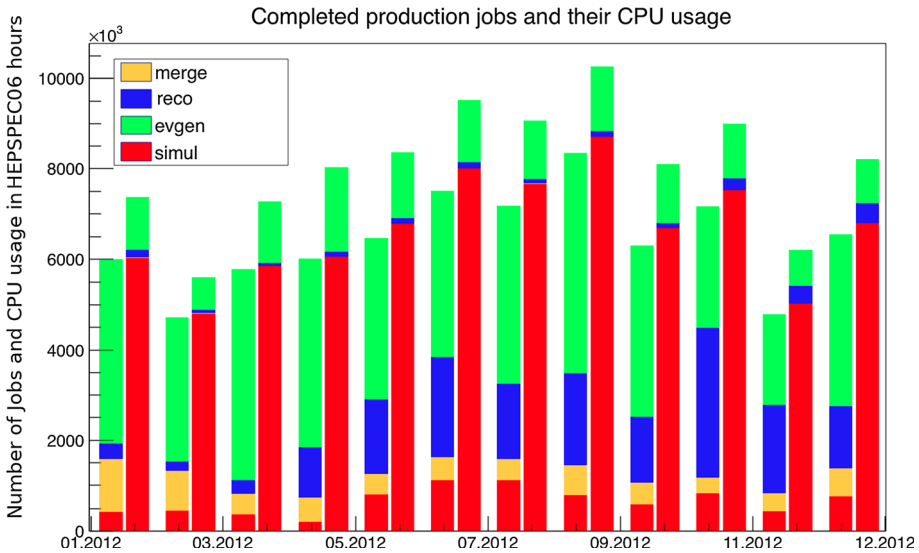


Fig. 1 Number of ATLAS jobs (*left bar*) per month and their CPU consumption (*right bar*) for 2012. The largest fraction of both the number of jobs and the overall CPU consumption is due to detector simulation jobs (simul). Also shown is the corresponding usage for event generation, merge and reco type jobs (numbers derived from [2])

indirectly, of computing resources. In the following sections we describe the basic principles behind JEM and give a brief description of its operation.

2 Replacing manual by automatic monitoring

The major steps involved in the production of physics simulations [1] at the LHC for its eventual use in physics analysis are depicted in Fig. 2. In the first two steps, events are generated using theoretical models and the particles produced are tracked in the detector volume; decays and interactions in the detector material are simulated. In a third step, the events are digitized, i.e. the response of sensors to the interacting particles is modeled. In all steps Monte Carlo (MC) techniques are used. Finally, physics objects are reconstructed from these signals and the resulting information is converted into a physics analysis data format.

In order to gather sufficient statistics, as well as to check for rare event topologies, tens of millions of events corresponding to a specific generator configuration have to be produced. The production of events with identical configurations is denoted in grid jargon as 'a task'. To minimize resource losses due to unexpected technical failures, each task is typically parallelized into many computing jobs with around 5000 events each. The tasks are processed serially, i.e. each step has to be completed for all generated events before the next step is initiated. This may lead to a significant latency between event generation and the final use of the samples in physics analysis. It also highlights the vulnerability of the production sequence to errors of any kind.

For example, problems introduced in the first step by faulty parameter settings may be identified only after all events have been produced and reconstructed.

For the most part collaborations rely on the effort of individual members to manually check numerous histograms in order to verify the integrity of the results. However, performing such validations typically involves a substantial effort, they are time consuming and require a high level of expertise. In fact, comprehensive checks consume so many resources that are actually prohibitive given the amount of information that needs to be validated. Additionally, monitoring 'by hand' suffers from an inherent delay in the availability of results until all checks are performed, and it is vulnerable to overlooking errors that are not obvious. It may happen that such problems are only detected after the simulation results have been used for several months in physics analyses when inconsistencies arise.

Instead of lengthy manual tests, which are prone to fail, the monitoring method described in this paper is based on two main characteristics:

– the checks are automated with a central tool,
– the checks are already performed during job execution in the grid.

Using such a method provides an immediate response to problems and allows one to monitor any kind of data quality information. The criteria to identify problems are unambiguously defined and agreed upon, i.e. it can be standardized. The proposed automated procedure needs to be developed only once and then it can be maintained centrally. Monitoring of production jobs can thus be significantly accelerated and simplified such that it provides an almost immediate response to problems. Furthermore, the list of observables to be monitored can easily be expanded. The procedure can be used to trigger the early termination of a job if serious problems are identified, thus preventing further waste of computing resources. The procedure is shown schematically in Fig. 2

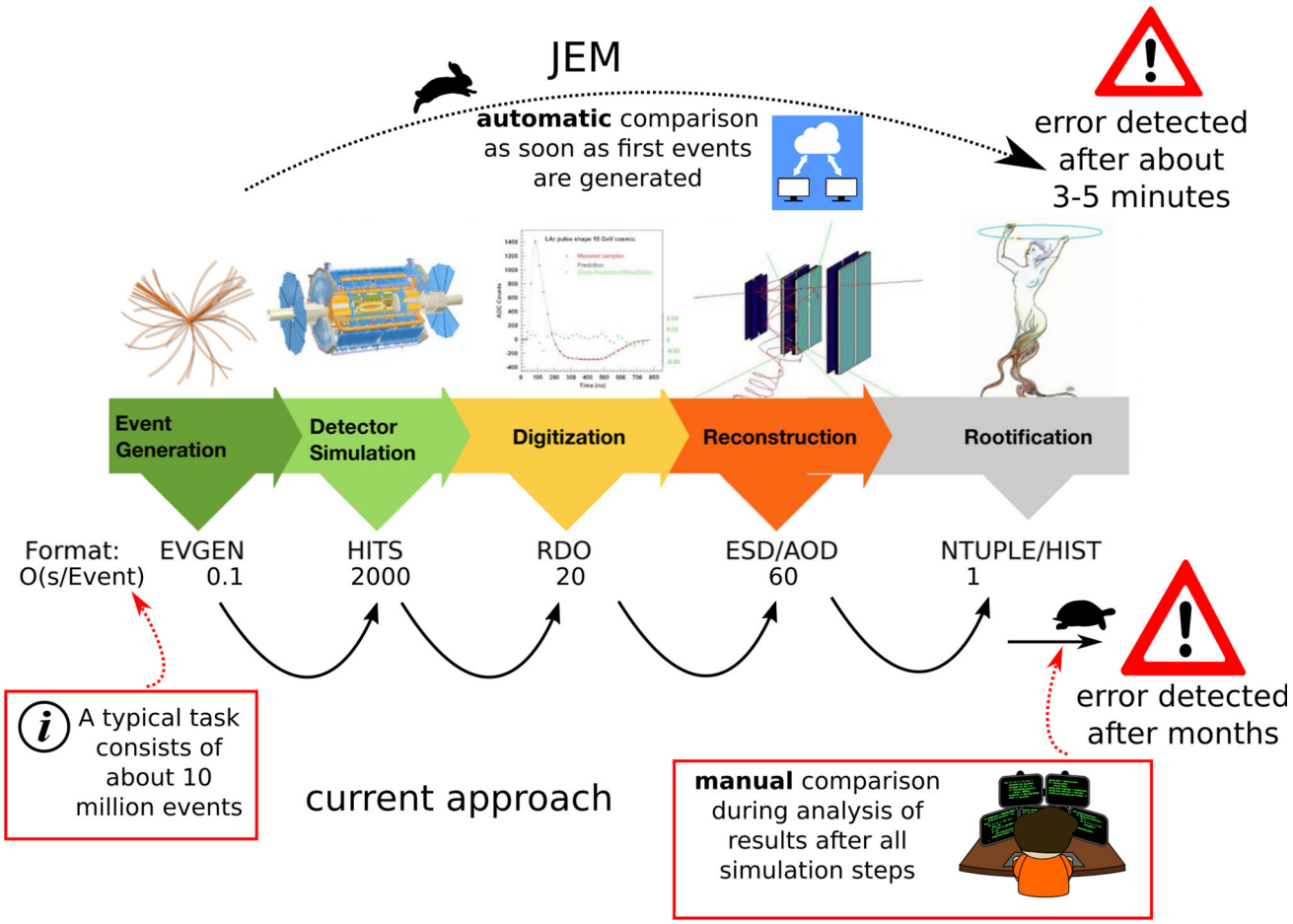


Fig. 2 Schematic representation of the steps involved in a simulation task, illustrating the advantages of the validation strategy discussed in this paper. In the current approach, if all simulation steps have completed successfully, possible problems can only be detected by chance during physics analysis. We propose here an automated validation process which is initiated at the very first step in the simulation sequence (diagram derived from [5])

for the case of monitoring in the generator step, where early monitoring has the highest payoff, as errors could be detected if present before the costly detector simulation step.

This new concept can be realized in JEM, which has been developed for online and remote data monitoring. While JEM has a wide range of features, in the context of this application, it uses a set of reference histograms and compares them to their respective counterparts produced by the grid task being monitored.

3 JEM

Before discussing its application to generator validation, the general concepts of the framework will be summarized. JEM was originally developed for online monitoring of jobs in a grid environment, where it aimed to identify causes of execution problems.

The job execution monitor has features to help to identify the reasons for job failures, or irregular job behavior, based on various metrics taken during the job runtime. These metrics include CPU consumption, memory and disk space, the number of open file descriptors and the network throughput. Advanced analysis techniques like online log file analysis and stack trace inspection are also available. An additional feature is the capacity to follow the step-by-step execution of Python and Bash scripts and of binary files that have been prepared before job submission to the grid.

JEM is mainly developed using the Python interpreter,[3] with a few core components written in the C programming language. The web services and the database backends are based on the Django framework [6].

In addition to the job instrumentation functionality, JEM provides several services at a central machine called the 'JEMserver'. These services include a database for monitoring job data storage, a web server to display monitoring and validation results and a central file cache.

[3] The Python part has grown to almost 230,000 lines of code.

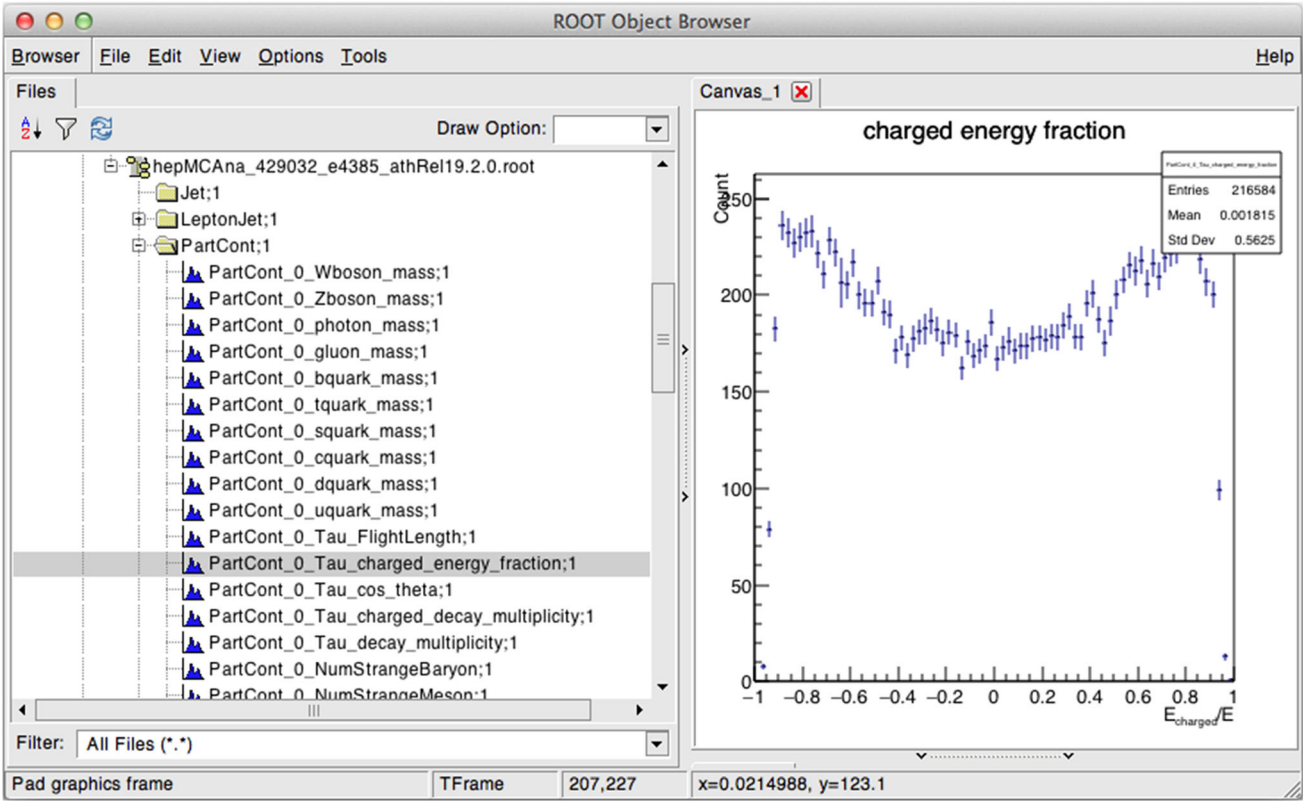


Fig. 3 ROOT browser showing parts of the topological structure of a HepMCAnalysis output file

4 Applying JEM to the validation of production

Perhaps the best usage potential of JEM is as an online data quality monitor. The validation technique supported by JEM is based on the comparison between two sets of histograms. The first set consists of certified histograms, which are derived from previous tasks and are denoted 'reference histograms.' These are stored in the central file cache and their specific definitions depend on the nature of the task to be validated. The second set of histograms is produced during the processing of the new task and is denoted 'quality histograms.' The production of these histograms and the statistical comparison of reference and quality histograms is performed automatically. In those cases where a process is known to be identical to an earlier reference, the evaluation of the result would be fairly trivial. However, in most use cases the parameters in the simulation have been changed (even if slightly), and the results would require the intervention of an expert to assess the severity of the possible disagreement. The detailed integration of these elements into JEM will be discussed next.

4.1 Categories of histograms

A key issue in the comparison of simulation tasks is the actual definition of the distributions to be compared. Until now, two standardized production frameworks for histogramming at the generator level have been invoked: HepMCAnalysis [7] and RIVET[4] [8]. Both produce ROOT [9] histogram collections organized into subdirectories defined by different analysis types. An example is shown in Fig. 3 for HepMCAnalysis. Expanding JEM validation to more analysis packages can easily be achieved by simply providing a small glue code.

4.2 Relevant JEM infrastructure

The JEM Activation Service is the kernel of the infrastructure (see Fig. 4) and it runs as a central service on the JEMserver. Every newly starting grid job contacts the JEMserver, which decides whether this particular job should be instrumented with JEM or not. In the current test version, this decision is steered manually by a web interface, and the monitoring jobs run just after the production task is finished. This service has been optimized for speed, database access is cached and the amount of exchanged data is very small. It is expected to support the load of about 11 jobs per second. Furthermore, a fallback procedure in every job prevents it from

[4] RIVET is a widely used analysis package in particle physics that contains comparisons of high energy physics data with MC simulations for a large number of processes and analyses.

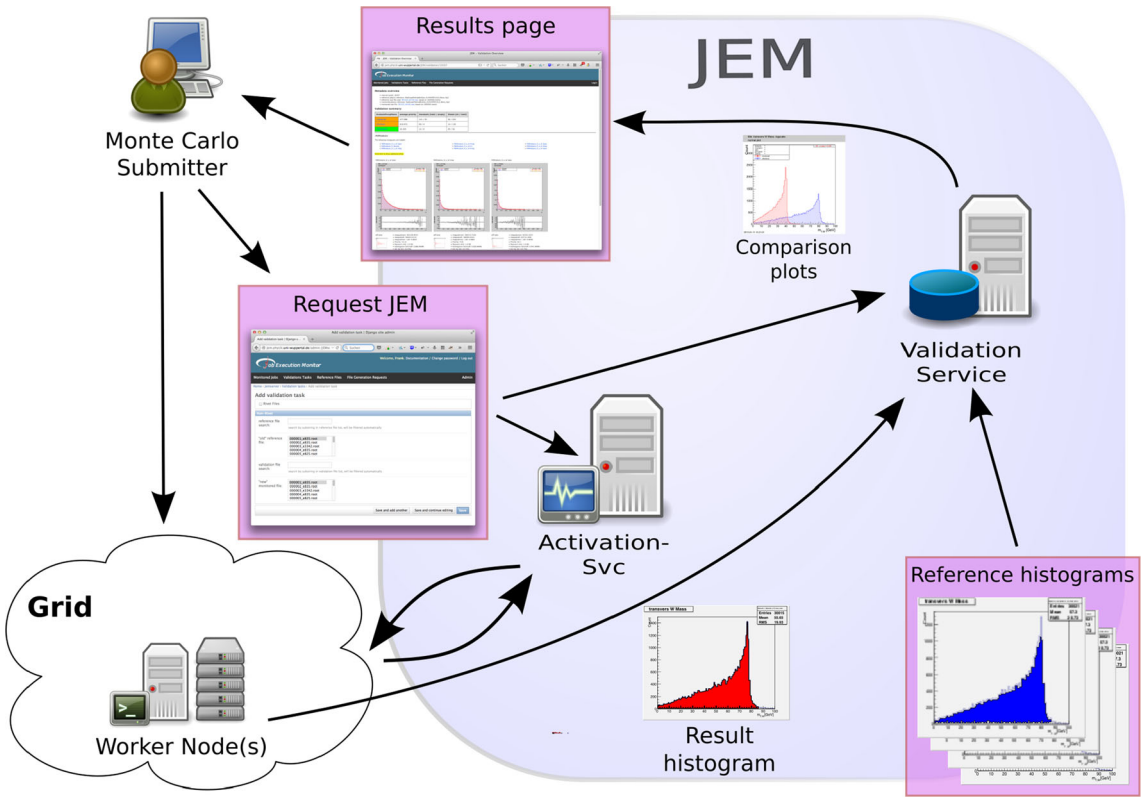


Fig. 4 Schematic representation of JEM's job validation in the grid. A Monte Carlo submitter sends a task to the grid and enters the task identifying data into the Request JEM web interface. From this interface, the Activation Service and the Validation Service are configured with the number of jobs to be instrumented and the choice of reference histograms to be used. Instrumented jobs produce quality histograms and send these to the Validation Service, where they are compared against the reference set. The comparison result can then be accessed via a web page

failing when the central server does not respond within a reasonable time frame. On average, around 200 jobs will be instrumented within each task to yield about a million events, providing sufficient statistical power to each histogram. The additional overhead in time needed to run the instrumented jobs is typically of the order of minutes, and therefore negligible compared to the total runtime of several hours needed for the simulation and reconstruction of events. A rule engine decides which job will be instrumented based on the given job metadata accompanying each request. This metadata includes, among other items, information as regards the job location, the submitter, the job type and the job identifier.

As the JEM monitoring component acts as a payload job's wrapper, it has full control over its child processes. Once it detects a successfully finished job, further actions can be initiated without notifying the outer batch system of the job's end. This is especially important for validation as will be described in Sect. 4.5.

The schematic workflow is shown in Fig. 4, and it is explained next.

4.3 Initiating JEM validation

The validation process within the JEM system is initiated using the JEMserver web interface.[5] This initialization step requires an authorized user with a valid account, which guarantees the typical data protection measures of individual experiments. The required input data includes the task identifier, the desired number of jobs to be instrumented and the keywords that select the respective physics analysis module.

The inputs are stored in a database and several services on the JEMserver either: add appropriate rules to the JEM Activation Service, or store a search pattern that is regularly executed to find a task matching the given criteria. The rules in the Activation Service keep track of the number of jobs that have been instrumented.

4.4 Quality histograms

Once jobs have been instrumented with JEM, quality histograms of the same type and format as the reference his-

[5] http://jem.physik.uni-wuppertal.de/.

Table 1 Metadata as stored in the database per file, both for reference and quality histogram files

Field	Description
fileName	Physical file name on disk
dataSetId	ATLAS specific; describing the physics process
eTag	Software tag: ATLAS specific; a tag associated with a software cache, containing specific software versions
swGenTag	Version of histogram generation software
swConfig	Configuration parameters for histogram generation software
physRef	Description of the underlying physics process
ts	Timestamp of creation or last update
eventCount	Number of events the file is based upon
generatedWith	Histogram generation software

tograms are created. This comprises more than 200 histograms. The production of quality histograms can be performed either online during job execution or offline after the instrumented jobs have finished.

Once the quality histograms have been created, they are sent to the JEMserver where they are merged and stored in the validation file cache. A database entry is added for each quality histogram collection file (an excerpt of its content is listed in Table 1). This file is also stored at the JEMserver and used as a reference file for future validations.

4.5 Validation process

The comparison between the monitored and the reference histograms is performed by the JEM Validation Service, which is a worker process running on the JEMserver. This service periodically checks if there are validation tasks in the database with a 'pending' state; when it finds one, it takes the oldest one and processes it. For visualization purposes and automatic quality control, a standard data quality program, DCube [10], is launched to perform a statistical analysis to estimate the level of agreement between the monitored and the reference file. This agreement is calculated using one or more statistical tests. Currently these are the Kolmogorov–Smirnoff test and Pearson's χ^2. Each of these statistical tests t returns a p value p_t. An overall estimator, the comparison severity S, is calculated using the weighted sum of the individual test results for each histogram comparison,

$$S = \frac{\sum_{\mathrm{t}}(1 - p_t) \cdot w_{\mathrm{t}}}{\sum_{\mathrm{t}} w_{\mathrm{t}}}, \quad (1)$$

where w_{t} is a weight factor that can be set to give preference to certain test results, and whose current default values are set to 1 for both tests. S is a simple measure of the inconsistency between both histograms. Histograms are considered consistent and labeled 'ok,' if $S < 0.25$. Values of $0.25 < S < 0.5$ would be labeled as warning, and values $S > 0.5$ are problematic and should be reviewed. It further allows one to present all histograms of an analysis group in an ordered fashion, e.g. having those with the largest S, i.e. the most inconsistent histograms, first.

4.6 Validation output web page

The results are stored in a file which includes all test results on the comparisons. In order to easily interpret results, they are presented graphically on a web page, an example of which is shown in Fig. 5. An overview of the metadata is given at the top. A summary of the quality of the validation is presented in classes according to the analysis groups. The information contains the average S value ('severity') and the number of non-empty and total histograms. The global agreement can easily be assessed since the outcome is presented using a color code of green, orange or red depending on the number of failed tests.

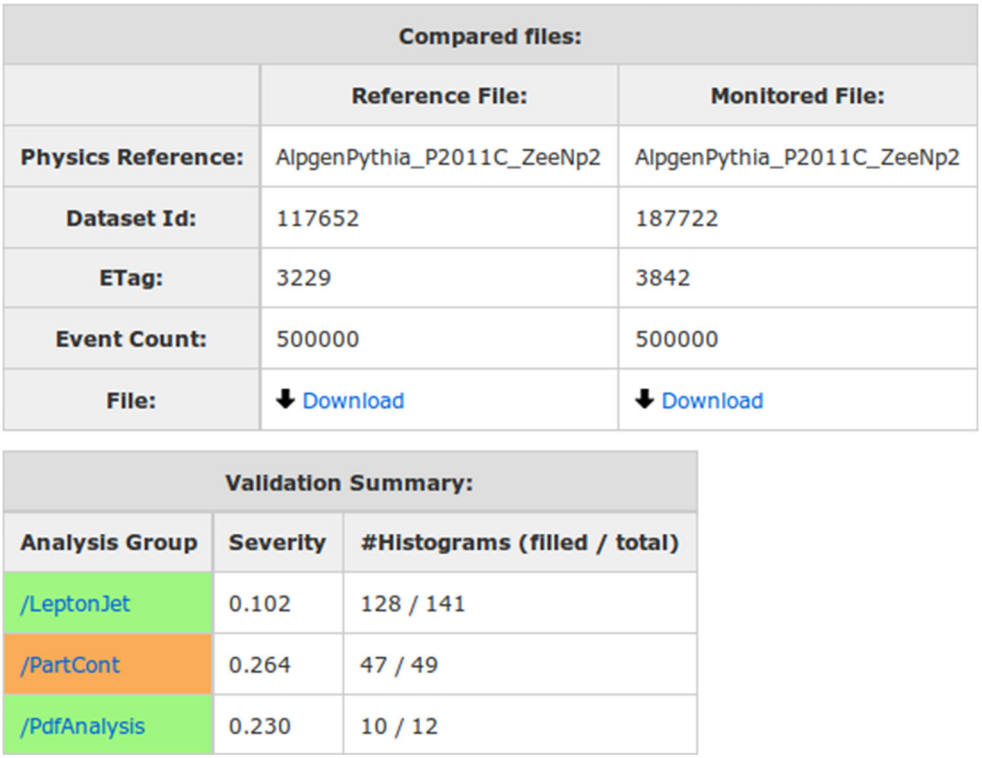
More detailed information on possible disagreement is provided through the plots of the individual distributions. The histograms for specific analysis modules are also listed (see Fig. 6). Within each module, they are ordered according to the magnitude of the severity S of the disagreement, with the histograms with the largest discrepancies displayed on the top. This ensures that the most problematic histograms can be spotted easily. A detailed picture of the (dis)agreement is possible since each quality histogram is overlaid over its reference histogram. Also included are the ratios between quality and reference histograms as well as the results of the statistical tests.

The fast turnaround and the immediate availability of the results on a web page provide an easy and fast tool for validation purposes. All necessary actions run automatically in the background and can also easily be configured using a web interface.

Although these validation steps run automatically, a review by an expert is often needed in order to assess the severity of differences whenever they appear. This manual evaluation is typically required given that tasks are rarely submitted under identical conditions; therefore, differences are expected to appear due to changes in parameter settings or to new versions of generators in need of validation.

5 Operational experience

In the two years since the first prototype of the JEM Validation Service was launched, more than 540 different task validations have been initiated by users of the ATLAS produc-



Compared files:		
	Reference File:	Monitored File:
Physics Reference:	AlpgenPythia_P2011C_ZeeNp2	AlpgenPythia_P2011C_ZeeNp2
Dataset Id:	117652	187722
ETag:	3229	3842
Event Count:	500000	500000
File:	Download	Download

Validation Summary:		
Analysis Group	Severity	#Histograms (filled / total)
/LeptonJet	0.102	128 / 141
/PartCont	0.264	47 / 49
/PdfAnalysis	0.230	10 / 12

Fig. 5 Top of a result web page. An overview of the metadata is given as well as a summary of the validation result. The *left column* of the lower table gives the name of the physics analysis, the center column lists the mean of the severity values of all histograms calculated using Eq. (1), leading to the color coding for each analysis group. The *right column* presents the number of filled and the number of total histograms produced. Histograms can be empty depending on the information accessible in the MC generator file and on the physics process investigated. Empty histograms are ignored for the calculation of the mean severity

tion system. This amounts to 197,660 individual histogram comparisons that have been evaluated. Overall the JEM Validation Service created 409 quality histogram files based on 345,703,764 events. As described in Sect. 4.1, two publicly available analysis software tools, HepMCAnalysis [7] and RIVET [8], are used to create the histograms which are actually employed in the comparison of the reference MC sample to the MC sample being validated. In both cases, JEM is very simple to use, prompting the ATLAS Collaboration to employ offline shift personnel to overview the validation, who required only a small amount of training. The majority of the tasks evaluated with JEM have shown no problematic distributions. The ease of interpreting results, e.g. via the color code scheme, has saved the MC validation shifters a significant amount of time and resources. This has made JEM crucial for putting out new simulation samples fast and reliably. In fact, a few mis-configured tasks were quickly discovered using JEM, which clearly showed them as problematic (some examples are given in Ref. [11]). JEM has also helped to quickly validate new versions of Monte Carlo generators, which is the topic of the next sections.

5.1 PYTHIA validation

One example involves the validation of PYTHIA 6.428 [12] which was compared to PYTHIA 6.427 as reference. This was the first validation campaign of PYTHIA in the ATLAS Collaboration using JEM. It became apparent that the reference version PYTHIA 6.427 had problems that were previously undiscovered. This shows that the systematic validation with JEM represents a large improvement compared to previous validation methods. The blue distribution in Fig. 7 shows the pseudorapidity distribution of the leptons that passed tight selection requirements in the Drell–Yan production, $pp \rightarrow e^+e^- + 2\,\text{partons} + X$. The two additional partons in the matrix element were generated using the ALPGEN generator [13]. To match parton showering and matrix element partons the MLM procedure [14] was invoked and the string fragmentation model was used to build primary hadrons from the final partons.

Apparently, the reference sample had a problem, because the distribution is not symmetric, in contrast to the expectation. The red distribution shows the same observable in the next PYTHIA version PYTHIA 6.428, showing that the

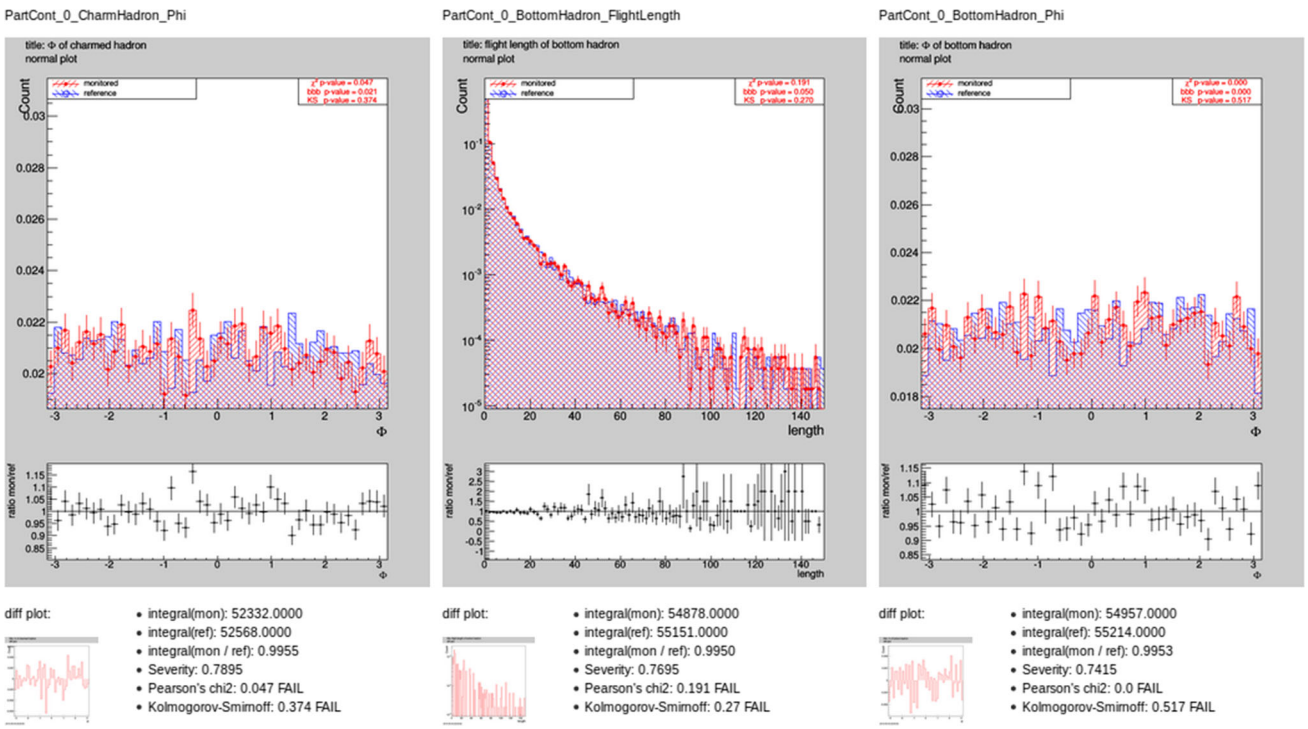


Fig. 6 Result web page of a specific analysis module containing the plots ordered by the severity S of the disagreement between quality and reference histograms. The ratio of the quality and the reference histograms is shown beneath each histogram. Also a small version of the difference plot is shown, which can be expanded to show the absolute difference between the two histograms. For each plot the statistical agreement is listed, together with the normalization scale. The three plots with the highest severity are shown together with their statistical test results. All other plots of an analysis group are hidden, but can be expanded on demand by clicking on the respective links

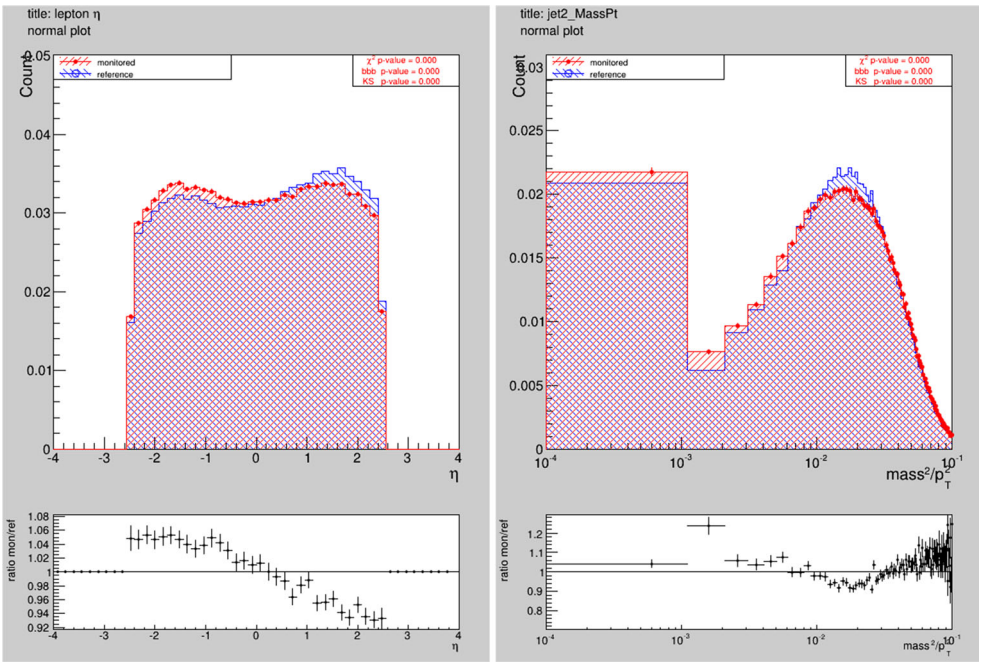


Fig. 7 Example distributions comparing ALPGEN+PYTHIA 6.427 (*blue*) with ALPGEN+PYTHIA 6.428 (*red*) in simulated $Z \rightarrow ee+2$ partons events. *Left* pseudorapidity distribution of tight electrons; *right* M^2/P_T^2 of the jets with second highest transverse momentum

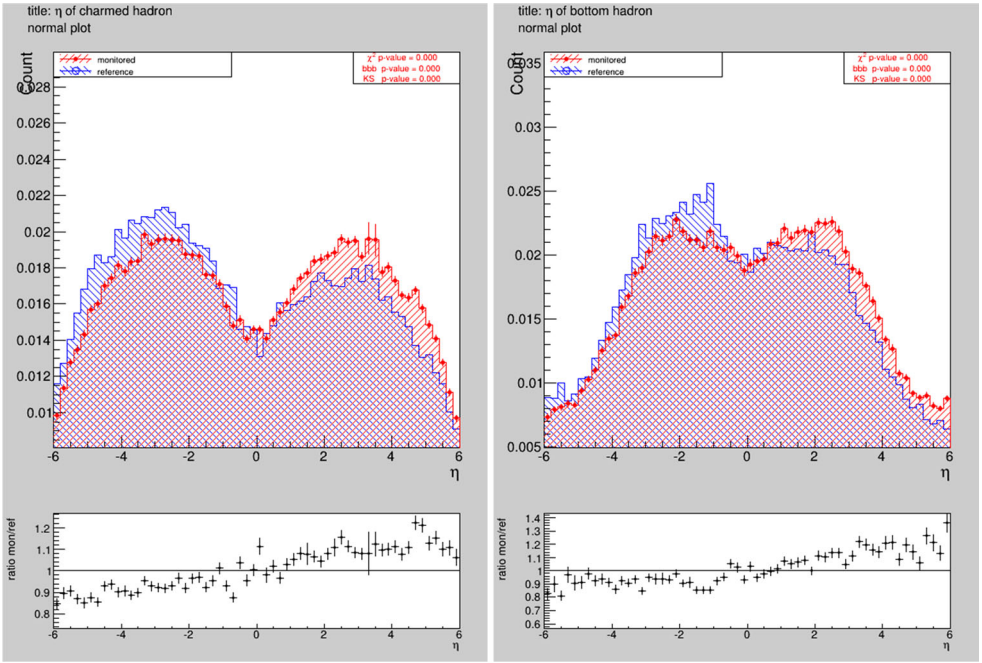


Fig. 8 Distribution of pseudorapidity of hadrons with charm (*left*) and bottom (*right*) quark flavor comparing SHERPA 2.1.0 (*blue*) with SHERPA 2.1.1 (*red*) for simulated $W \rightarrow e\nu+$ jets events

problems were fixed in this version. ATLAS only discovered this error after investigating systematically with JEM. Consequently, the version PYTHIA 6.427 was discarded from further use.

On the other hand, JEM was also used to quickly check if intended changes in the distributions were actually realized in the results. An example from the two versions of PYTHIA is shown in Fig. 7 (right). Here the distribution of the jet mass squared M^2 over p_T^2 is displayed. This shows that both versions differ in the simulation, a result that was expected since for the newer version some changes in the PYTHIA string fragmentation simulation were implemented by the authors.

5.2 SHERPA validation

The validation campaign of SHERPA [16] provides yet another example of the discovery of a previously unknown problem during a systematic MC validation with JEM.

SHERPA 1.4 is a multi leg generator containing tree-level matrix elements. For the $W+$ jets production process this involves the multi-parton matrix elements for $W+$ 0 jets, $W+$ 1 jet, $W+$ 2 jets, and $W+$ 3 jets in leading order (LO) quantum chromodynamics (QCD), combined with the QCD parton cascades following the CKKW merging scheme [15], and fragmentation of partons into primary hadrons described using a phenomenological cluster-hadronization model.

SHERPA 2.1.0 is one of the first publicly available versions of this multi leg generator that also included matrix elements in next-to-leading order (NLO) QCD. In case of $W+$ jets production, this version contains matrix elements of $W+$ 0 jets, $W+$ 1 jet, and $W+$ 2 jets in NLO QCD, and $W+$ 3 jets, $W+$ 4 jets in LO QCD, matched to parton showers using the CKKW method. This version represents a major step in developing MC multi leg generators including NLO QCD matrix elements. Since the changes to previous versions are large, a very careful MC validation was necessary. The systematic MC validation method using JEM proved essential in this task.

During the validation of SHERPA 2.1.0, problems with the generator soon became obvious. Generating $W+$ jets events with subsequent $W \rightarrow e\nu$ decay in proton–proton scattering at a center-of-mass energy of 8 TeV, for example, displayed asymmetric distributions of the pseudorapidity of hadrons including charm or bottom flavor, while those distributions were expected to be symmetric as predicted in SHERPA 1.4. After private communication with the SHERPA authors, this problem could be traced to an error in the multiple parton interaction matrix elements, which was fixed in version SHERPA 2.1.1.

This can be seen in Fig. 8 where the pseudorapidity distributions for charm hadrons (left) and for bottom hadrons (right) are compared between the repaired version SHERPA 2.1.1 (red) and the problematic version SHERPA 2.1.0 (blue). As a result, in version SHERPA 2.1.1, the pseudorapidity distributions of charm and bottom hadrons are now symmetric.

The validation campaign of SHERPA 2.1.0 represents another success story for JEM. It was established then that a prompt identification of issues within a new release of generators is possible with this tool. The information rendered by

JEM and the large set of plots it produced were of tremendous help to the SHERPA authors to fix this issue and other problems.[6] The prompt discovery of the issue discussed above avoided the mass production of mis-configured samples by the ATLAS and CMS Collaborations, thus saving an enormous amount of CPU resources.

6 Summary and outlook

In this paper we present a new validation tool based on the job execution monitor (JEM). It runs in the grid framework as a job payload during event generation, automatically collecting and comparing data quality measures of simulated physics processes. JEM's current test version (which allows manual steering through a web interface) has already become a standard validation tool in ATLAS simulation, giving users the opportunity to identify faulty results due to mis-configurations in Monte-Carlo generators and/or unexpected features of simulation tasks. It accomplishes this task by automatically classifying and comparing certified reference histograms against those produced during the simulation task being monitored. This procedure helps to identify problems early enough, preventing the waste of computing resources in mis-configured jobs. Furthermore, it has proven to be extremely helpful in validating new versions of QCD generators.

JEM's user interface consists of a web front-end where results are presented. The web service uses a simple color scheme to signal the level of agreement with the reference task. Histograms are displayed sorted from highest to lowest severity, making it possible to quickly assess the overall status of the monitored simulation task. This interface was greatly improved in collaboration with the ATLAS Monte Carlo working group and their shifters through extensive hands-on testing. It was demonstrated that the overhead of both, the generation of validation plots and the automatic comparison of histograms, was negligible compared to the runtime of typical simulation tasks.

ATLAS is currently investigating the possibility of instrumenting automatically a small fraction of every generator task, making quality histograms readily available for their inspection should the necessity arise at a later time.

Although JEM is currently being used only in the early generator step of the simulation chain, its validation system could be extended to later steps, such as the simulation, digitization (or reconstruction) step and beyond, where it would also provide significant savings in time and computing resources. The simplicity of the JEM installation and operation would also make it possible to export its use outside of ATLAS and HEP experiments.

[6] In a similar way, for example, a problem in the shower simulation of partons produced close to the beam axis was found which led to a modification in the next SHERPA 2.2.0 version.

Acknowledgements We would like to thank Leonid Serkin and Maria Moreno Llacer, together with the ATLAS generator validation team, for their invaluable feedback on using the JEM Validation Service. It helped to improve the overall structure and user experience. We are grateful to Markus Elsing and Marcelo Vogel for very helpful comments on the text. Also, we would like to thank Frank Ellinghaus for discussions, Ewelina Lobodzinska for the computing support and Nataliia Kondrashova for the HepMCAnalysis modules.

Funded by SCOAP[3].

References

1. G. Aad et al., The ATLAS simulation infrastructure. Eur. Phys. J. C **70**, 823 (2010)
2. E. Karavakis et al. [ATLAS Collaboration], Common accounting system for monitoring the ATLAS distributed computing resources. J. Phys. Conf. Ser. **513**, 062024 (2014)
3. T. dos Santos. New approaches in user-centric job monitoring on the LHC computing grid. Ph.D. thesis, Bergische Universität Wuppertal, 2011
4. T. dos Santos et al., Application of remote debugging techniques in user-centric job monitoring. J. Phys. Conf. Ser. **368**, 012012 (2012)
5. C. Debenedetti, Concepts for fast large scale Monte Carlo production for the ATLAS experiment. ATL-SOFT-SLIDE-2013-790, Oct 2013. http://cds.cern.ch/record/1605835
6. The Web framework for perfectionists with deadlines|Django. 2014. https://www.djangoproject.com
7. S. Johnert. Measurement of the $W \to \tau \nu\_\tau$ cross section in proton–proton collisions at ATLAS and the development of the HePMCAnalysis tool. Ph.D. thesis, Hamburg U., 2012. http://www-library.desy.de/cgi-bin/showprep.pl?thesis12-009
8. A. Buckley et al., Rivet user manual. arXiv:1003.0694 [hep-ph]
9. R. Brun et al. ROOT—an object oriented data analysis framework. In Proceedings AIHENP'96 Workshop, Lausanne, Sep. 1996, Nucl. Inst. and Meth. in Phys. Res. A 389, 1997. See also http://root.cern.ch/
10. DCube Documentation, 2014. https://twiki.cern.ch/twiki/bin/view/Sandbox/DCubeDoc
11. ATLAS Collaboration, Validation of Monte Carlo event generators in the ATLAS Collaboration for LHC Run 2, Jan 2016, ATL-PHYS-PUB-2016-001. http://cds.cern.ch/record/2119984
12. T. Sjostrand, S. Mrenna, P.Z. Skands, JHEP **0605**, 026 (2006)
13. M.L. Mangano, M. Moretti, F. Piccinini, R. Pittau, A.D. Polosa, JHEP **0307**, 001 (2003)
14. M.L. Mangano et al., Nucl. Phys. B **632**, 343 (2002). arXiv:hep-ph/0108069
15. S. Catani, F. Krauss, R. Kuhn, B.R. Webber, JHEP **0111**, 063 (2001). doi:10.1088/1126-6708/2001/11/063. arXiv:hep-ph/0109231
16. T. Gleisberg et al., J. High Energy Phys. **02**, 007 (2009)